

# A Small-Scale Research Platform for Intelligent Transportation Systems

Hung Manh La, Ronny Salim Lim, Jianhao Du, Weihua Sheng, Gang Li, Sijian Zhang and Heping Chen

**Abstract**— In this paper, we propose and develop a small-scale research platform for intelligent transportation systems. Our platform has four main parts: an arena; an indoor localization system; automated radio controlled (RC) cars; and roadside monitoring facilities. First, to mimic the traffic environments we build an arena with a wooden floor, mock buildings and streets. Second, for the indoor localization system, a motion tracking system (Opti-Track) is set up to track the RC cars for control purpose. Third, for the automated RC cars, both manually and automatically controlled RC cars are developed. The automatic one is equipped with a micro controller, an Xbee RF module, a microphone and a speaker. The manual RC car is similar to the automatic one but equipped with a small wireless camera. The designed circuit inside the RC cars allow them to: (1) receive control signal from the computer through Xbee, and (2) control the front and rear wheels through motors. The control algorithm is developed to allow the RC car to track predefined trajectories. Fourth, we develop the roadside monitoring facilities, which consists of an IP-based fish-eye camera and the associated video processing modules including image segmentation, object identification and tracking. Several experiments are conducted to demonstrate the effectiveness of the designed platform.

**Keyword:** Intelligent transportation systems, Dynamic tracking, Non-holonomic vehicle, RC car.

## I. INTRODUCTION

With nearly 43,000 deaths a year on U.S. roads [1], a need exists for countermeasures to reduce the number and severity of traffic accidents. Intelligent Transportation Systems (ITS) become more and more important in recent years since they can reduce accidents, save life and traveling time, especially in urban areas. Currently there are many areas of ITS that are under development, aiming at the reduction of vehicle accidents, the realization of automatic driving, the relief of traffic congestion and the improvement of the environment [2].

ITS has attracted many researchers around the world [1], [3]. The overview of main issues, technological challenges, developments, and achievements of ITS can be found in [4]. Panwai *et al.* [5] presented the findings from a comparative evaluation of car-following behavior in a number

This project is supported by Oklahoma Transportation Center grant OTCREOS10.1-43 and NSF grant CISE/CNS MRI 0923238.

Hung Manh La is with the Center for Advanced Infrastructure and Transportation, Rutgers, the State University of New Jersey, Piscataway, NJ 08854, USA, email: hung.la11@rutgers.edu

Ronny Salim Lim, Jianhao Du, Weihua Sheng and Gang Li are with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078, USA, emails: (ronny.lim, jianhao.du, weihua.sheng, gang.li)@okstate.edu

Sijian Zhang is with the College of Electrical Engineering, Zhejiang University, Hangzhou, Zhejiang 310027, China, email: zsjzju@yahoo.com.cn

Heping Chen is with the Ingram School of Engineering, Texas State University, San Marcos, TX 78666, USA, email: hc15@txstate.edu.

of traffic simulators such as advanced interactive microscopic simulator for urban and rural networks, and parallel microscopic simulation. Furthermore, autonomous vehicles such as Googles self-driving cars and fully automated Buick LeSabres vehicles have been developed [6], [7]. Moreover, the Integrated Vehicle-Based Safety Systems (IVBSS) have been developed and integrated in a fleet of 16 passenger cars and 10 heavy trucks. Their goal is to examine the effect of the prototype of the IVBSS integrated crash warning system on driving behavior and driver acceptance [8], [9].

However, there are still many challenges in conducting full scale ITS research. For example testing in real traffic environments is usually dangerous. Modifying from conventional vehicles to fully autonomous ones is usually costly in terms of money and time. These reasons motivate us to build a small scale testbed for ITS research that can simulate real traffic environments and driving experience. This small scale testbed will provide features such as easy to access, easy to duplicate and low cost. Therefore it will allow many researchers to conduct research in ITS.

The rest of this paper is organized as follows. In the next section we present the hardware setup of the intelligent transportation system testbed. Section III shows the hardware design for the automated RC car. Section IV describes the RC car model, the tracking control algorithm, and the control of multiple RC cars. Section V presents the vision based monitoring of traffic. Section VI shows the experimental results. Finally, Section VII concludes this paper.

## II. HARDWARE SETUP OF THE INTELLIGENT TRANSPORTATION SYSTEM

### A. Overall System Setup

In this section we present our hardware setup for the platform of the intelligent transportation system. Our platform has four main parts:

- An arena,
- An indoor localization system,
- Automated radio controlled (RC) cars,
- Roadside monitoring facilities.

To mimic the traffic environments we build an arena with a wooden floor, mock buildings and streets. An indoor localization system built from a motion tracking system (Opti-Track) is developed. Automated radio controlled cars including both manually and automatically controlled prototypes are developed and tested. An affordable hardware platform consisting of an off-the-shelf RC car, a micro-controller, a microphone, a speaker and a mini wireless camera (for manually controlled RC car only) is designed.

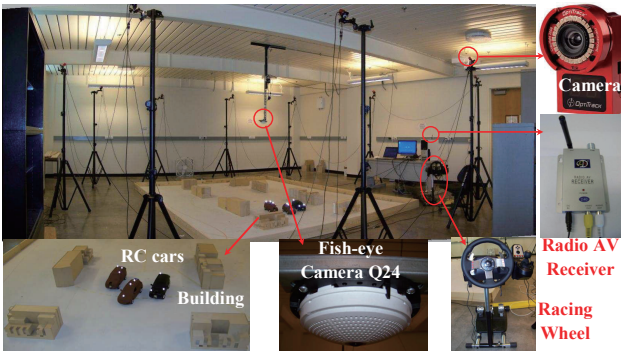


Fig. 1. The overall platform for ITS research.

The designed circuit inside the RC cars allow them to receive the control signal from the computer through Xbee wireless communication, and control the front wheels based on a built-in servo motor and the rear wheels based on a built-in DC motor. For the roadside monitoring facilities, an IP-based fish-eye camera (Q24) associated with advanced video processing modules including image segmentation, object identification and tracking is developed.

### B. Indoor localization system

An indoor localization system (see Figure 1) is built up to localize on RC cars in the simulated traffic environment. The basic purpose of this system is to provide location and orientation feedback of the cars in order to control them to move along predefined tracks, for example, commanding four cars move across the traffic intersection without collision. The system consists of an optical motion capture system (Opti-Track) from NaturalPoint Inc [10], and a PC. There are 12 cameras in the Opti-Track system fixed around the arena so that the whole area of the arena is covered. The Opti-Track is fast enough to capture 100 frames per second, so that location and orientation information can be obtained in real time and high accuracy. The Opti-Track system tracks each RC car via the markers (see Figure 2) mounted on the top of the car. According to the predefined tracks stored in the PC, the control algorithm generates the right commands to control the speed and the motion (e.g. “move forward”, “backward”, “turn left” and “turn right” or “stop”) of each car.

### C. Automated RC cars

We have developed both automatically and manually controlled RC cars.

For the automatically controlling RC car platform (see Figure 2-Top), the markers are mounted on the top of the RC car to build a rigid body in order to get location and orientation of the car. The speaker is mounted on the back of this car for the purpose of crash-sound playback. The tracking control algorithm that allows the RC car to track predefined trajectories is developed in the computer, and the control signal is sent to the RC car via the Xbee communication.



Fig. 2. Two RC car platforms: (Top) for automatic driving control, (Bottom) for manually driving control.

For the manually controlled RC car platform (Figure 2-Bottom), to mimic human driving, a small wireless camera is mounted on the hood of the RC car to observe the environment in front of the car and broadcast the video stream through wireless communication. The video stream is sent to the receiver, then displayed on the monitor. The human driver sits in front of the wheel stand and controls the RC car while he observes the video stream. The whole setup of such a manually driving system is shown in Figure 3.



Fig. 3. The setup for manually driving the RC car.

In Figure 3 the steering wheel is used to control the RC car in order to mimic human driving behavior. We developed a C++ program using the software development kit (SDK) from Logitech to read the status data of the steering wheel including turning angle, brake, gas pedal and gear shift. Then, we use these data to control the RC car, such as “move forward”, “backward”, “turn left”, “turn right”, or “stop” through Xbee wireless communication. We use two different channels (different frequencies) for the wireless

communication in order to avoid data collision between the Xbee module and the wireless camera.

#### D. Roadside monitoring facilities

An IP-based fish-eye camera (Q24) as shown in Figure 1 is mounted over the arena. This camera is capable of providing different views simultaneously as well as a full 360 degree all-round view, hence it can cover the whole area of the arena to monitor the RC cars. This camera uses an Ethernet-based interface. The stream of live images from the camera is obtained by setting up a socket connection. The features of the camera (including resolutions, frame rates, etc) can be easily modified by sending a web request. Also the zooming and panning of the camera lenses can be done using virtual PTZ function. The camera provides a highest resolution at 3M pixels and the color images are scalable from  $160 \times 120$  to  $2048 \times 1536$ . The highest frame rate is about 30fps. In the future, more sensors will be deployed to further enhance the monitoring system.

### III. HARDWARE DESIGN OF AUTOMATED RC CAR

In this section we present the hardware design of the automated RC car. The RC cars used in our system are based on commercial off-the-shelf parts. The scale of these cars is 1:14. Originally the RC car has a front DC motor and a rear DC motor. In order to obtain a controllable orientation and wider steering angles, the front DC motor is removed, and a servo motor is mounted instead. The overall hardware design of our RC car control system is shown in Figure 4.

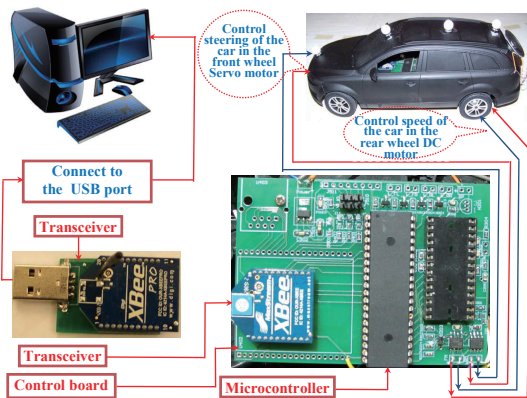


Fig. 4. Full hardware setup for the RC car control system.

Below we will explain the Xbee wireless module and the control board.

*Xbee wireless module:* The module is using small, low-power digital radio based on the IEEE 802.15.4 standard for wireless personal area networks (WPANs). In this module, there are a transmitter connected to the PC and a receiver on the control board as shown in Figure 4. The data is streamed between them with serial communication protocol. Through the Xbee module, the wireless communication is established between the PC and RC cars.

*Control board:* An embedded control board is developed to replace the original circuit board inside the RC car. The

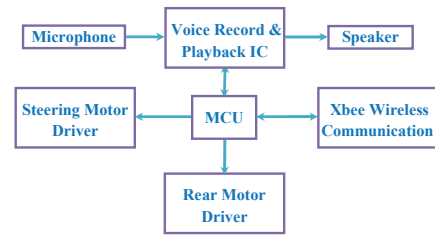


Fig. 5. The function blocks of the control board.

function blocks of the control board is shown in Figure 5. The MCU (ATmega 162) in the middle of Figure 5 functions as a brain for the automated RC car. A microphone is used to record collision sound while a speaker is used to playback. Both the microphone and the speaker are driven by a voice record IC (ISD 1700). The microphone and the speaker can be used to mimic the crash sound. The PWM output from the MCU is used to drive the front servo motor and rear DC motor to adjust the orientation and the velocity of the RC car, respectively.

### IV. RC CAR CONTROL

The problem of controlling a non-holonomic vehicle is well studied [11]. However, controlling a non-holonomic vehicle with low accuracy and non-smooth velocity is a challenging problem. In this section, we build a model of the RC car, then focus on developing efficient control algorithms for the RC car to track a predefined trajectory. In addition, we develop multi-thread programming for multi-car control.

#### A. RC Car Model

As we know a common model of non-holonomic vehicles is usually described as:

$$\begin{cases} \dot{x} = v \cos(\Psi) \\ \dot{y} = v \sin(\Psi) \\ \dot{\Psi} = \omega \end{cases} \quad (1)$$

here  $\Psi$  is the orientation angle of the vehicle (see Figure 6), and  $\omega$  is the angular velocity. The model in (1) is simple and does not consider the actual constraints on the range of steering angle and the sliding angle which reflects the sliding errors between the center point of the car and the center point of the front axial.

Since the RC car has low accuracy on steering we model it as:

$$\begin{cases} \dot{x} = v \cos(\Psi + \theta + \beta) \\ \dot{y} = v \sin(\Psi + \theta + \beta) \\ \dot{\Psi} = \omega \end{cases} \quad (2)$$

here  $\theta$  is the steering angle of the front wheels (see Figure 6), and  $\beta$  is the sliding angle that is obtained based on the center point of the car and the velocity vector  $v$ . The  $\beta$  angle is computed as  $\beta = \Psi_c - \Psi$ , here  $\Psi_c$  is the heading of the vehicle at the center point.

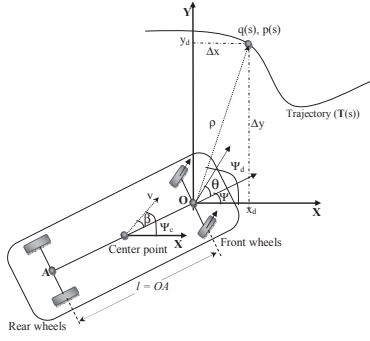


Fig. 6. Illustration of the mobile platform (non-holonomic vehicle) tracking the virtual vehicle ( $p(s)$ ,  $q(s)$ ) moving in an arbitrary trajectory  $T(s)$ .

### B. RC Cars Control Algorithm

In order to let the RC car track a predefined trajectory we use a virtual vehicle based approach. The virtual vehicle,  $s(t)$ , is designed to move on the path with  $x_d = p(s)$  and  $y_d = q(s)$ . In order to track the virtual vehicle, two constraints are considered in inequalities (3) and (4), which are related to the difference between the actual heading and the desired heading of the RC car, and the distance between the actual and the virtual RC car, respectively.

$$\lim(|\Psi(t) - \Psi_d(t)|)_{t \rightarrow \infty} \leq d_\Psi, \quad (3)$$

here  $\Psi_d$  is the desired angle,  $d_\Psi$  is a small angle threshold.

$$\lim(\rho(t))_{t \rightarrow \infty} \leq d_\rho, \quad (4)$$

here  $d_\rho$  is a small distance threshold.  $\rho(t)$  is the Euclidean distance between the RC car and the virtual vehicle (see Figure 6). It is computed as:

$$\rho(t) = \sqrt{\Delta x^2 + \Delta y^2}, \quad (5)$$

here  $\Delta x = x_d - x$  and  $\Delta y = y_d - y$ .

In order to handle the inequality (3), the steering angle control for the RC car is based on the proportional- derivative control (PD control) which is as follows:

$$\theta = -k_p(\Psi - \Psi_d) - k_d(\dot{\Psi} - \dot{\Psi}_d),$$

here  $k_p$  and  $k_d$  are positive constants.

As mentioned in the previous section, although we replaced the front DC motor by a servo motor to obtain a controllable orientation and wider steering angles (20 degree), the left and right steering angles are not the same because the mechanical steering part is not rigid. Additionally, the velocity of the RC car is not smooth, and the steering angle can not be accurately controlled.

The low accuracy steering angle of the RC car is illustrated in Figure 7. In this model, since the front wheels mounted on the car are not stable, the left steering angle range (left in Figure 7) is different from the right one (right in Figure 7). Namely, in our experiment we use an RC car which has  $\theta_L^{max}$  between  $22^\circ$  and  $27^\circ$  and  $\theta_R^{max}$  between  $15^\circ$  and  $20^\circ$ . This difference on the left and right steering angle is one of the reasons that make the car not able to track the

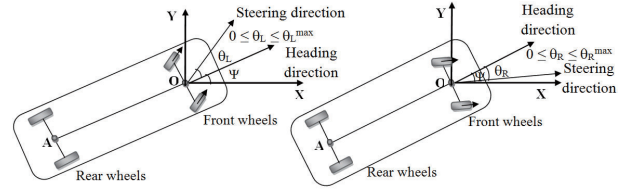


Fig. 7. Low accuracy steering angle of the RC car. The left steering angle range is different from the right one.

predefined trajectories when we apply traditional tracking control algorithms.

To design a new control algorithm, we introduce a virtual vehicle. Our idea here is if the car moves slowly then the virtual vehicle has to wait for it by decreasing its velocity gradually. Otherwise the virtual vehicle move in smooth velocity for the RC car to track it.

Based on the above analysis, in order to handle the inequality (4) the parameter  $\gamma$  is introduced [12], [13]

$$\dot{\rho} - \dot{d}_\rho = -\gamma(\rho - d_\rho), \quad (6)$$

here  $\gamma$  is a positive constant.

From (5) we can obtain:

$$\begin{aligned} \dot{\rho} &= \frac{1}{\sqrt{\Delta x^2 + \Delta y^2}} (\Delta x \dot{x} + \Delta y \dot{y}) \\ &= \frac{1}{\rho} [\Delta x (\dot{x}_d - \dot{x}) + \Delta y (\dot{y}_d - \dot{y})], \end{aligned} \quad (7)$$

here  $\dot{x}_d = \dot{p}(s)\dot{s}$  and  $\dot{y}_d = \dot{q}(s)\dot{s}$ .

Substituting  $\dot{\rho}$  from (7) into (6) with attention that  $d_\rho$  is a constant,  $\dot{d}_\rho = 0$ , we have

$$\frac{1}{\rho} \dot{s} (\Delta x \dot{p}(s) + \Delta y \dot{q}(s)) = \frac{1}{\rho} (\Delta x \dot{x} + \Delta y \dot{y}) - \gamma(\rho - d_\rho) \quad (8)$$

or,

$$\dot{s} = \frac{1}{\Delta x \dot{p}(s) + \Delta y \dot{q}(s)} [(\Delta x \dot{x} + \Delta y \dot{y}) - \gamma(\rho - d_\rho)] \quad (9)$$

From (9) it is easy to see that if  $\Delta x \dot{p}(s) + \Delta y \dot{q}(s) = 0$  then  $\dot{s} \rightarrow \infty$ , or the car can not track the virtual vehicle. In order to have  $\Delta x \dot{p}(s) + \Delta y \dot{q}(s) \neq 0$  the car should stay close to and behind the virtual vehicle. From this analysis we should make the virtual vehicle move with constant velocity at initial time when the car is far from it. This means that we need to design  $s(t) = s(t-1) + c$  ( $c$  is a positive constant) if  $t < t_{threshold}$ .

Finally the whole tracking control algorithm is shown in the Algorithm 1.

### C. Multi-car Control

For multi-car control, the control algorithm is implemented using multi-thread programming. The workstation has a USB wireless adapter (XBee) installed for communicating with the RC cars. Each car has an XBee receiver on it. The structure of the multi-thread control program is shown in Figure 8. All RC cars are controlled by the same workstation. However, the controller for each car is implemented independently in a

---

**Algorithm 1: Virtual Vehicle Tracking Algorithm**


---

**Initialization phase:**

- Create a trajectory of the virtual vehicle that the RC car wants to track.
- Initialize parameters:  $v, k_p, k_d, d_\rho, \gamma, \Delta_t, s, \dot{s}, \Delta_t$ .

**Implementation phase:**
**if**  $t < t_{threshold}$  **then**

 Let the virtual vehicle move with constant velocity  
 (to relax the assumption  $\Delta x \dot{p}(s) + \Delta y \dot{q}(s) = 0$ )

$$s(t) = s(t-1) + c,$$

 here,  $c$  is a positive constant.

**else**

- Compute the velocity of the virtual vehicle:

$$\dot{s}(t) = \frac{1}{\Delta x \dot{p}(s) + \Delta y \dot{q}(s)} (\Delta x \dot{x} + \Delta y \dot{y} - \gamma \rho (\rho - d_\rho)).$$

- Compute the steering angle for the RC car:

$$\theta = -k_p(\Psi - \Psi_d) - k_d(\dot{\Psi} - \dot{\Psi}_d).$$

- Update the position of the virtual vehicle based on its velocity  $\dot{s}$ :

$$s(t) = s(t-1) + \dot{s}(t)\Delta_t.$$

**end**


---

separate thread. The decision making for each car is handled locally in these RC car threads. These threads also handle the communication with the cars. The main program thread connects to the cars and initializes the other threads.

In our experiment, each thread receives the location and orientation of the corresponding RC car from the OptiTrack system. The virtual vehicle is also defined in the thread, so each car has its virtual vehicle. After the tracking computation is completed, each thread gives an output to control the car by writing the data in a global variable. The main thread is used to send the value of the global variable to all RC cars.

For the wireless communication, we use one Xbee module to control multiple RC cars by sending a package to all RC cars. This package contains the data to command all RC cars. The Xbee modules on the RC cars run in the same channel. In this project we use three cars to run autonomous tracking. The sent information contains the data of velocity and steering to control the individual RC cars. We also investigate that the wireless communication can not guarantee that the client Xbee receives the correct data. To solve this problem, we apply a simple filter in the microcontroller to remove the bad data. Since we assign an ID for each RC car, the transmitted data also contains the ID. By matching the ID of the RC car and the data, all cars can be controlled using a single Xbee. The multi-car control approach is shown in Figure 8.

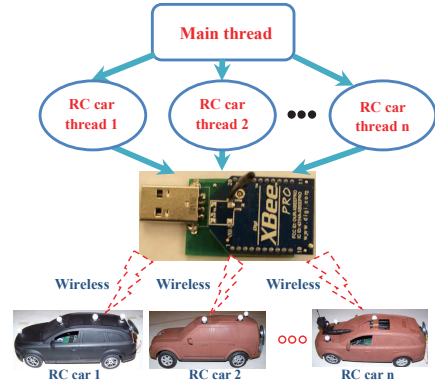


Fig. 8. The structure of the multi-thread control program.

## V. VISION BASED MONITORING OF TRAFFIC

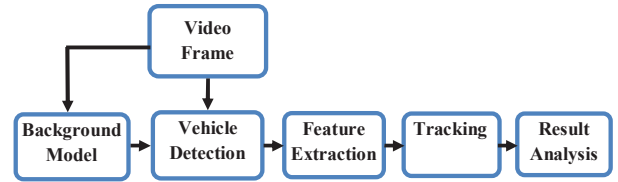


Fig. 9. The flowchart of the vision tracking process.

In this section we present the vision-based monitoring of traffic. Main applications of traffic monitoring are vehicle detection and tracking. An IP-based fish-eye camera (Q24) is chosen to cover the whole area of the arena for monitoring. We set the camera with a resolution of  $800 \times 600$  pixels which ensures the real-time processing. Based on the live image sequences received from the fish-eye camera (Q24) via the Ethernet, the vision processing of detecting and tracking the RC cars is done remotely in a PC, and the processing speed is around 8fps.

The flow chart is shown in Figure 9. Firstly a threshold-based background model is applied to obtain foreground from image sequences for the purpose of detecting the motion regions of the cars. After the regions are detected, color features are extracted to label different cars and then the tracking algorithm, Kalman filter, is applied to get more accurate information about the car trajectories.

The system (state) model is defined as:

$$\begin{cases} x(t) = x(t-1) + \Delta x(t-1) \\ y(t) = y(t-1) + \Delta y(t-1) \\ \Delta x(t) = \Delta x(t-1) \\ \Delta y(t) = \Delta y(t-1) \end{cases} \quad (10)$$

here  $(x(t), y(t))$  is the location of the car at time  $t$ .

The observation model is defined as:

$$\begin{cases} z_x(t) = x(t) + v_x(t) \\ z_y(t) = y(t) + v_y(t) \end{cases} \quad (11)$$

here  $(z_x(t), z_y(t))$  is the observation of the location of the car at time  $t$ , and  $v = [v_x \ v_y]^T$  is Gaussian and uncorrelated noise with zero mean and variance of one.

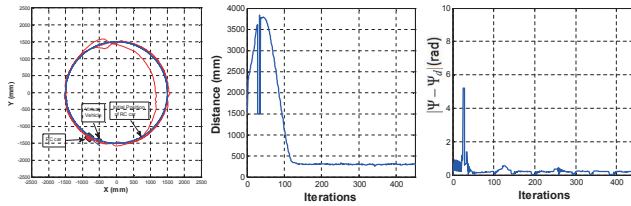


Fig. 10. Trajectories of the RC car tracking the virtual vehicle moving in circle (Left), the distance between the RC car and the virtual vehicle (Middle), and the difference between the real heading of the RC car and the desired one (Right).

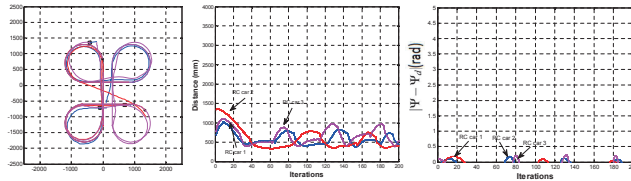


Fig. 11. Trajectories of the 3 RC cars tracking the virtual vehicles moving in desired trajectory (Left), the distance between the RC car and the virtual vehicle (Middle), and the difference between the real heading of the RC cars and the desired ones (Right).

The vision information is obtained and saved, and it can be used to verify the trajectory and velocity information determined by RC car control system. The vision algorithm is implemented in C/C++ using OpenCV.

## VI. EXPERIMENTAL RESULTS

In this section we test our proposed control algorithm for a single RC car, then we extend the test to three RC cars based on multi-thread programming. We also provide the results of car tracking using the vision tracking algorithm.

- Parameter setup:  $v = 67$ ,  $k_p = 1$ ,  $k_d = 0.8$ ,  $d_p = 300mm$ ,  $\gamma = 2$ ,  $\dot{s}(0) = 0$ ,  $\Delta_t = 0.00056$ .

- Parameters of the virtual vehicle moving in a circle:  $[x, y] = [R\cos(s), R\sin(s)]$  with  $R = 1500mm$ .

The tracking results of Algorithm 1 are shown in Figures 10. Namely, Figure 10 (Left) shows the RC car tracking the virtual vehicle which moves in the circle trajectory. Figure 10 (Middle) shows the evaluation of the distance between the RC car and the virtual vehicle, and we can see that this distance converges to the predefined value of  $d_p = 300mm$ . Hence, this result satisfies the inequality (4). In addition, we evaluate the difference between the real heading of the RC car and the desired one as shown in Figure 10 (Right). This result also satisfies the requirement in (3).

We also evaluate Algorithm 1 for 3 RC cars tracking. Figure 11 (Left) shows the three RC cars tracking the three virtual vehicles which move in the mock streets with an intersection. Figure 11 (Middle) and (Right) show the evaluations of the distance between the RC cars and the virtual vehicles, and the difference between the real heading of the RC cars and the desired ones, respectively.

For vehicle detection and tracking through the overhead Q24 camera, we first let two RC cars track the predefined trajectories based on Algorithm 1. One tracks the virtual

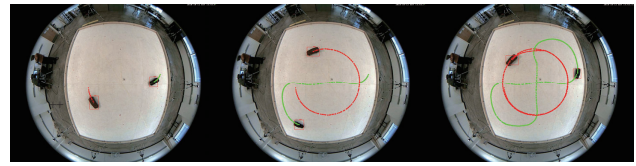


Fig. 12. Snapshots of vision tracking algorithm.

vehicle moving in circle trajectory, and other one tracks another virtual vehicle moving in figure “8” trajectory. Then, the vision tracking algorithm is applied to track these two cars. The snapshots of the tracking result are shown in Figure 12.

## VII. CONCLUSION

This paper proposes a small scale research platform for the intelligent transportation system. Four main parts of the platform: an arena, an indoor localization system, automated RC cars and roadside monitoring facilities are developed. Along with this new platform development, this paper studies the problem of controlling a non-holonomic vehicle which has low steering accuracy and non-smooth velocity. The tracking control algorithms for such a vehicle are proposed. Experimental results are collected to demonstrate the working of this ITS research platform and the RC car control performance. We believe this platform can be used in studying many ITS related problems.

## REFERENCES

- [1] J. Njord, J. Peters, M. Freitas, B. Warner, K. C. Allred, R. Bertini, R. Bryant, R. Callan, M. Knopp, L. Knowlton, C. Lopez, and T. Warne. Safety applications of intelligent transportation systems in Europe and Japan. *International Technology Scanning Program*, pages 1–54, 2006.
- [2] S. Hollborn. Intelligent transport systems (ITS) in Japan. *Technical report, Technische Universitat Darmstadt*, pages 1–38, 2002.
- [3] L. Li, X. Li, Z. Li, D. D. Zeng, and W. T. Scherer. A bibliographic analysis of the iee transactions on intelligent transportation systems literature. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):251–255, 2010.
- [4] T. G. Crainic, M. Gendreau, and J.Y. Potvin. Intelligent freight-transportation systems: Assessment and the contribution of operations research. *Transportation Research Part C*, 17:541–557, 2009.
- [5] S. Panwai and H. Dia. Comparative evaluation of microscopic car-following behavior. *IEEE Transactions on Intelligent Transportation Systems*, 6(3):251–255, 2010.
- [6] Google. Autonomous driving. *The New York Times*, Oct 10, 2010.
- [7] Vehicle platooning and automated highways. *California Path*, pages 1–4, 2010.
- [8] Integrated Vehicle-Based Safety Systems (IVBSS). <http://www.umtri.umich.edu/divisionpage.php?pageid=249>.
- [9] J. R. Sayer, S. E. Bogard, M. L. Buonarosa, D. J. LeBlanc, D. S. Funkhouser, S. Bao, A. D. Blankespoor, and C. B. Winkler. Integrated vehicle-based safety systems, light-vehicle field operational test key findings report. *Technical Report, The University of Michigan Transportation Research Institute*, pages 1–132, 2011.
- [10] <http://www.naturalpoint.com/optitrack/>.
- [11] J. Cochran and M. Krstic. Nonholonomic source seeking with tuning of angular velocity. *IEEE Transactions on Automatic Control*, 54(4):717–731, 2009.
- [12] M. Egerstedt, X. Hu, and A. Stotsky. Control of mobile platforms using a virtual vehicle approach. *IEEE Transactions on Automatic Control*, 46(11):1777–1782, 2001.
- [13] S. V. Gusev and I. A. Makarov. Stabilization of program motion of transport robot with track laying chassis. *Proceedings of LSU*, 1, 1989.